International Journal of Engineering Research and Development
***e*-ISSN: 2278-067X, *p*-ISSN: 2278-800X, www.ijerd.com**
Volume 3, Issue 7 (September 2012), PP. 15-19

A critical Comparison of Graph Clustering Algorithms Using the K-clique Percolation Technique

Mousumi Dhara [1], K.K.Shukla[2], Neeraj Sharma[3]
[1]*Research Scholar, Department of Computer Engineering, IIT-BHU, Varanasi*
[2]*Professor, Department of Computer Engineering, IIT-BHU, Varanasi*
[3]*Associate Professor, School of Biomedical Engineering, IIT-BHU, Varanasi*

Abstract*—In the recent past, various graph clustering algorithms have been proposed. Each algorithm has its own behaviour in terms of performance on a specific data set. So, it is really hard to tell which one is the most efficient and optimal. The concept of k-clique percolation technique in random networks is introduced where k is the size of the complete sub-graphs that are organized into large scale cluster and are analytically and numerically investigated. Erdos-Renyi random graph which is undirected and unweighted is choosen for studying the k-clique percolation technique. In an Erdos-Renyi graph with N vertices, where two vertices are connected to each other by an edge with probability $p_c(k)$, the percolation transition of kcliques takes place when $p_c(k) = [(k-1)N]^{-1/(K-1)}$. Clique percolation has been used in the past for identifying overlapping communities in large real networks. In this paper, restricted neighbourhood search clustering (RNSC) and Markov Clustering (MCL) algorithms are tested in terms of efficiency and optimality using Erdos-Renyi random graphs with varying graph sizes and also using the k-clique percolation technique. The comparison is done between these two algorithms in terms of cluster size and run-time with varying p_c (k) according to increasing graph size. To validate the cluster quality obtained by these algorithms, normalized mutual information (NMI) and adjusted mutual information (AMI) are calculated using large scale Erdos-Renyi graph. It is shown that RNSC algorithm is better than MCL using the k-clique percolation technique in terms of cluster size, run-time, NMI and AMI.

***Keywords*— RNSC, MCL, Run-time, Cluster size, NMI, AMI**

I. INTRODUCTION

Cluster analysis has been in great interest in wide class of complex systems, occurring from the level of cells to society. The real networked data are used for the analysis of performance measure in terms of anomalous degree distribution, diameter, spreading phenomena, clustering coefficient and correlations [1, 2]. The local structural unit of networks has been paid great attention in the field of graph clustering. The global features can be interpreted by the distribution and clustering properties [3] of small and well defined sub graphs, introduced as "motifs" [4].Communities [5] are often represented as somewhat larger units, made up of vertices, which are more densely connected to each other than to the rest of the network and also denoted as the most essential structural units of real networks. In recent past, many graph clustering methods like Markov Clustering (MCL) [6], restricted neighbourhood search clustering (RNSC) [7], and Molecular Complex Detection (MCODE) are used to divide the whole network into smaller pieces .But the most important concept of overlapping communities is not introduced in those methods. Percolation is a technique on graphs which can be applied to identify the overlapping communities effectively and deterministically in large real networks. The uncorrelated classical Erdos–Renyi random graph [8] is related to the percolation transition [9]. The transition is taking place at $p=p_c \equiv 1/N$, where p is the probability that two vertices are connected by an edge and N is the total number of vertices in the graph. A giant component which is also referred to as the percolating component is appeared and resulted in a dramatic change in the overall topological features of the graph and also in the centre of interest for other networks as well.

In recent past RNSC, MCL came into the field of graph clustering and each algorithm has it's own method and relies on a very different approach. RNSC, which is a cost based clustering method and performs some local search iteratively to obtain some optimum clustering in an efficient way. RNSC algorithm is a stochastic technique, which uses restricted neighbourhood search. The memory requirement for RNSC is o (n^2).The worst-case memory required is o (n^2) when the graph is completely connected. The complexity of a move in the naive cost function is o (n), which is the size of the restricted neighbourhood of a move M. The Markov clustering, proposed by Stijn van Dongen, this delivers a very fast clustering method and also provides a natural clustering in weighted graphs [10]. This algorithm is based on the prototype of stochastic flow simulation technique. So it is very tough to say that which one is the most efficient and optimal in the case of performance and robustness. Here, using the percolation technique it can be checked that which one gives better performance in terms of efficiency and optimality between RNSC and MCL.

II. GRAPH CLUSTERING ALGORITHMS AND PERCOLATION TECHNIQUE

A. RNSC (Restricted neighbourhood search Clustering)

RNSC [11] is a local search metaheuristic technique. It is used to minimize the cost function in the solution space. According to Stijn van Dongen, the vertex-wise performance criteria for clusterings on unweighted graphs are as the sum of coverage measure is taken on each vertex. When the naive cost function fails in considering changes to small neighbourhood

as being more significant than changes to large neighbourhoods then a more expressive new cost method, scaled cost is derived. The scaled function tries to optimize the output from naive function and reach to the global optimal solution.

B. MCL (Markov clustering)

The Markov clustering, proposed by Stijn van Dongen, which is provided a very fast clustering method and is also contributed a natural clustering in weighted graphs. The prototype of stochastic flow simulation technique is used for that algorithm. Flow expansion and inflation operators are used to produce a natural grouping of densely flow-connected vertices. These two operators are obtained from the input graph and transformed the probability of the random walk in the markov chain like way to another. Actually, the inflation is strengthening the flow where it is strong and also weakening the flow where it is already weak and the flow expansion is used for propagating the flow within the graph. MCL is very fast also for sparse graphs.

C. Clique percolation (Percolation technique)

The analytic and simulation results are related to the appearance of a giant component, made of complete subgraphs of k vertices (k-cliques) and those results are obtained through the calculation of the threshold probability. Some concepts of k-clique adjacency, k-clique chain, k-clique connectedness and k-clique percolated subgraph are introduced here: i) K-clique adjacency: two k-cliques are adjacent if they share k-1 vertices, i.e., if they differ only in a single vertex ii) K-clique chain: a subgraph which is the union of a sequence of adjacent k-cliques. iii) k-clique connectedness: two k-cliques are k-clique –connected if they are parts of a k-clique chain. iv) k-clique percolation cluster (component): it is a maximal k-clique connected subgraph, i.e., it is the union of all k-cliques that are k-cliques connected to a particular k-clique.

In the k-*clique adjacency graph*, k-clique percolation cluster is very much like a regular (edge) percolation cluster where the vertices represent the k-cliques of the original graph, and there is an edge between two vertices if the corresponding k-cliques are adjacent. *Rolling* a k-*clique template* is very much like moving a particle from one vertex of this adjacency graph to another one along an edge is equivalent to from one k-clique of the original graph to an adjacent one. A k-clique template can be assumed of as an object that is isomorphic to a complete graph of k vertices. Such a template can be located onto any k-clique of the original graph, and rolled to an adjacent k-clique by repositioning one of its vertices and keeping its other k−1 vertices fixed. Thus, the k-clique percolation clusters of a graph are all those subgraphs that can be fully discovered but cannot be left by rolling a k-clique template in them. Now, the threshold probability (critical point) of k-clique percolation using heuristic arguments is presented as a general result. It is examined that a giant k-clique component appears in an Erdos-Renyi at $p = p_c(k)$, where

$$Pc(k) = \frac{1}{[(k-1)N]^{\frac{1}{K-1}}}. \qquad (1)$$

Obviously, for k = 2 this result agrees with the known percolation threshold ($p_c = 1/N$) for Erdos-Reyni graphs, because 2-clique connectedness is equivalent to regular (edge) connectedness.

D. Some parametric concepts

NMI (Normalized Mutual Information): It is the measure of the quality of clusters, which is the mutual information shared between clusterings. This is originally proposed by Alexander Strehl and Joydeep Ghosh [12]. Assume, there are set of groupings of clusterings as $\{\lambda^{(q)} \mid q \in \{1,..,r\}\}$ which is denoted by $\wedge$. Let $n_h^{(a)}$ be the number of objects in cluster c_h according to $\lambda^{(a)}$ and $n_l^{(b)}$ be the number of objects in cluster c_l according to $\lambda^{(b)}$. Let $n_{h,l}$ represents the number of objects that are in c_h according to $\lambda^{(a)}$ and in cluster c_l according to $\lambda^{(b)}$.The symbol $\phi^{(NMI)}$ is denoted as the estimation of NMI.

$$\phi^{(NMI)}(\lambda^{(a)}\lambda^{(b)}) = \frac{\sum_{h=1}^{k^{(a)}}\sum_{l=1}^{k^{(b)}} n_{h,l}\log(\frac{n.n_{h,l}}{n_h^{(a)}n_l^{(b)}})}{\sqrt{(\sum_{l=1}^{k^{(b)}} n_l^{(b)}\log(\frac{n_l^{(b)}}{n}))(\sum_{h=1}^{k^{(a)}} n_h^{(a)}\log(\frac{n_h^{(a)}}{n}))}} \qquad (2)$$

AMI (Adjusted Mutual Information):

Table 1: The Contingency Table, $n_{ij} = (U_i \cap V_j)$

U\V	V_1	V_2	...	V_C	Sums
U_1	n_{11}	n_{12}	...	n_{1C}	a_1
U_2	n_{21}	n_{22}	...	n_{2C}	a_2
⋮	⋮	⋮	⋱	⋮	⋮
U_R	n_{R1}	n_{R2}	...	n_{RC}	a_R
Sums	b_1	b_2	...	b_C	$\sum_{ij} n_{ij} = N$

To correct the measures for randomness it is necessary to specify a model according to which random partitions are generated. Such a common model is the "permutation model" [13], in which clusterings are generated randomly subject to having a fixed number of clusters and points in each clusters. Using this model, which was also adopted by Hubert and Arabie (1985) for the ARI, we have previously shown [14] that the expected mutual information between two clusterings **U** and **V** is:

$$E\{I(U,V)\} = \sum_{i=1}^{R}\sum_{j=1}^{C}\sum_{n_{ij}=\max(a_i+b_j-N,0)}^{\min(a_i,b_j)} \frac{n_{ij}}{N}\log\left(\frac{N.n_{ij}}{a_i b_j}\right)\frac{a_i!b_j!(N-a_i)!(N-b_j)!}{N!n_{ij}!(a_i-n_{ij})!(b_j-n_{ij})!(N-a_i-b_j+n_{ij})!} \tag{3}$$

As suggested by Hubert and Arabie (1985), the general form of a similarity index corrected for chance is given by:

$$\text{AdjustedIndex} = \frac{\text{Index- Expected Index}}{\text{Max Index-Expected Index}}$$

This is upper-bounded by 1 and equals 0 when the index equals its expected value. Having calculated the expectation of the MI, the adjusted form is proposed, which is denoted as the *adjusted mutual information* (AMI), for the normalized mutual information. For example, taking the NMI*max* we have:

$$\text{AMI}max(\mathbf{U},\mathbf{V}) = \frac{\text{NMImax(U,V)-E\{NMImax(U,V)\}}}{\text{1-E\{NMImax(U,V)\}}} = \frac{\text{I(U,V)-E\{I(U,V)\}}}{\max\{\text{H(U),H(V)}\}\text{-E\{I(U,V)\}}} \tag{4}$$

III. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this section, a number of experiments with different test cases are presented to evaluate the efficiency, cluster size and optimality of graph clustering algorithm RNSC, compare to MCL with Erdos-renyi graph data. There are 15 experiments done with tabu length 50 for obtaining the result in case of RNSC and in case of MCL there is using highest inflation value 3 with scheme 5 for obtaining the result.

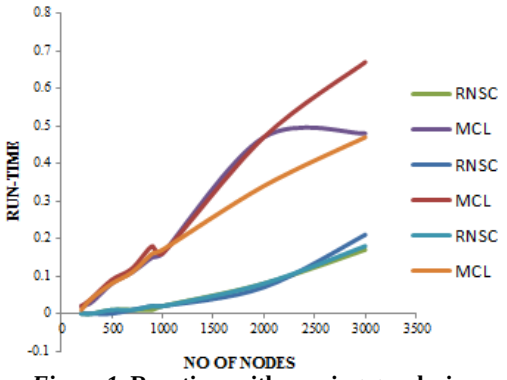

Figure 1. **Run-time with varying graph size**

From Fig.1, it is observed that the run-time is decreasing in case of RNSC in three conditions when $p_e = p_c$, $p_e < p_c$ and $p_e > p_c$ and in case of MCL, run-time is increasing gradually in the three cases for k= 6 cliques community.

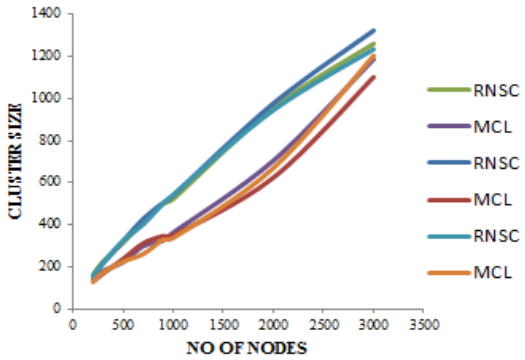

Figure 2. **Cluster size with varying graph size**

From Fig.2, it is observed that the cluster size is growing larger for RNSC in these three cases $p_e = p_c$, $p_e < p_c$ and $p_e > p_c$ but in case of MCL, cluster size is gradually decreasing in the three cases for k=6 cliques community.

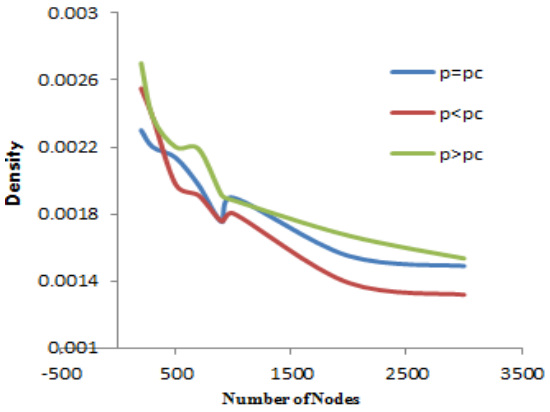

Figure 3. **Density with varying graph size**

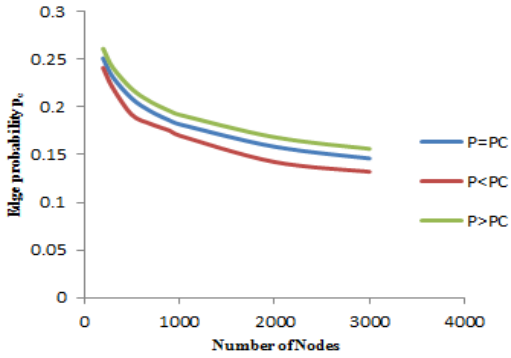

Figure 4. **Edge probability p_e with varying graph size**

From Fig.3,Fig.4, it is observed that the density and probability measure for the Erdos Renyi graph of different graph size in the three conditions when $p_e = p_c$, $p_e < p_c$ and $p_e > p_c$ for k=6 cliques community.

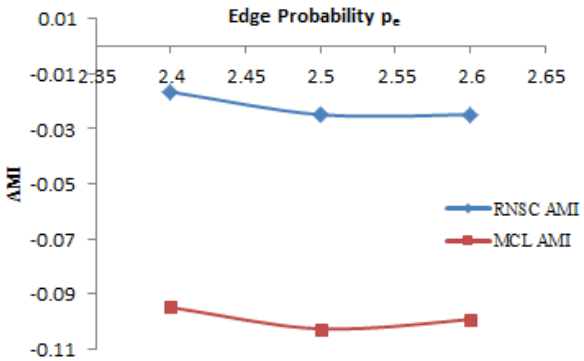

Figure 5. **AMI with edge probability P_e**

From Fig.5, it is revealed that the AMI measure of both RNSC and MCL's optimality check on the basis of edge probability p_e for k=6 cliques community .In this figure RNSC is performing better than MCL for the three cases as $p_e < p_c$, $p_e = p_c$ and $p_e > p_c$.

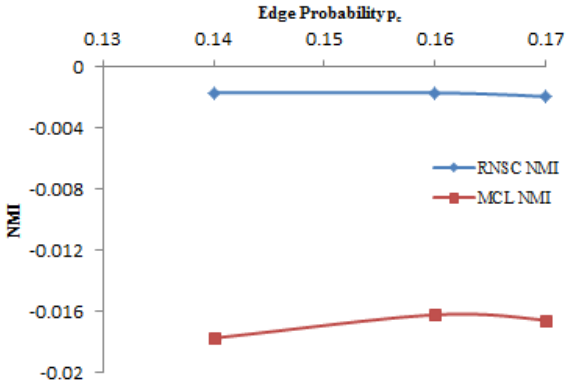

Figure 6. **NMI with edge probability p_e**

From Fig.6, it is revealed that the NMI measure of both RNSC and MCL's optimality check on the basis of probability of k=6 cliques community. In this figure also, RNSC is performing better than MCL for these three cases as $p_e < p_c$, $p_e = p_c$ and $p_e > p_c$.

IV. CONCLUSIONS

This paper presents a critical comparison between two popular graph clustering algorithms RNSC and MCL using percolation technique in terms of efficiency and optimality .The results show that RNSC is more efficient and optimal than MCL in the three cases as $p_e = p_c$, $p_e < p_c$ and $p_e > p_c$. Using Erdos-Renyi graph with different graph sizes, all the testings are done for RNSC and MCL algorithms. NMI and AMI are used to evaluate the optimal clustering. RNSC and MCL algorithms are tested using Erdos-Reyni graphs with different graph sizes. The time complexity of RNSC is o (n^3).The time complexity of MCL is o ($n.k^2$) where n is the number of nodes and k is the number of resources allocated per node. Also RNSC can be further extended by parallel move method which will give better result in the case of run-time or average cost.

REFERENCES

[1]. D. J. Watts and S. H. Strogatz, Collective dynamics of 'small-world' networks, Nature **393**, 440, 1998.
[2]. A.-L. Barab´asi and R. Albert, Emergence of Scaling in Random Networks, Science **286**, 509, 1999.
[3]. A. Vazquez, R. Dobrin, D. Sergi, J.-P. Eckmann, Z. Oltvai and A.-L. Barab´asi, Subgraphs and networks motifs in geometric networks, cond-mat/0408431.
[4]. R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii and U. Alon, Network Motifs: Simple Building Blocks of Complex Networks, Science **298**, 824,2002.
[5]. M. Blatt, S. Wiseman, and E. Domany, Superparamagnetic Clustering of Data, Phys. Rev. Lett. **76**, 3251, 1996.
[6]. S. M. van Dongen, Graph Clustering by Flow Simulation. PhD thesis, University of Utrecht, 2002.
[7]. Andrew Douglas King, Graph Clustering with Restricted Neighbourhood Search, M.S Thesis, University of Toronto,2004.
[8]. P. Erd˝os and A. R´enyi, on the evolution of random graphs, Publ. Math. Inst. Hung. Acad. Sci. **5**, 17, 1960.
[9]. Imre Der´enyi, Gergely Palla, and Tam´as Vicsek, Clique percolation in random networks, submitted to Phys. Rev. Lett., *Budapest, Hungary, 2005.*
[10]. S. M. van Dongen , A cluster algorithm for graphs,Technical Report INS-R0010, Centrum voor Wiskunde en Informatica,2002.
[11]. Andrew King, An Efficient Cost-Based Graph Clustering Algorithm, University of Toronto, 2005.
[12]. Strehl, A. and Ghosh,J. , Cluster ensembles - a knowledge reuse framework for combining partitionings, AAAI, 93–98,2002.
[13]. H.O Lancaster.,The chi-squared distribution. New York, John Wiley, 1969.
[14]. V. Singh, L. Mukherjee, J. Peng, and J. Xu. , Ensemble clustering using semidefinite programming with applications. *Mach. Learn.*, doi: 10.1007/s10994-009-5158-y, 2009.